Few terms from my days in graduate school continue to spark more controversy than software engineering. Some believe that we’ve matured as a community to the point that we have enough rigor in our processes to justify having the term “engineering” next to the word “software.” Others contend that software development is still an art form in search of a true engineering foundation. Because both arguments have merit, the debate continues, and likely will for many years to come.

In this special issue, we leave that “lightning rod” debate alone and instead look at the relationship between core business principles and software engineering.

**What you need to know**

The IEEE Computer Society’s Software Engineering Body of Knowledge covers a lot of territory in its 10 key knowledge areas:

1. Software requirements
2. Software design
3. Software construction
4. Software testing
5. Software maintenance
6. Software configuration management
7. Software engineering process
8. Software engineering tools and methods
9. Software engineering management
10. Software quality

As you can see, you need a lot of expertise to be well-versed in these disciplines. But
that’s only the knowledge and technology piece of the puzzle. Software engineering is also a business, and it’s a business whose results increasingly run other businesses.

Regardless of these 10 knowledge areas, in practice, software engineering in an organization often looks more like a loosely coupled family of ad hoc techniques, standards, tools, practices, methodologies, and people. But somehow, when it’s used, this nonphysical entity behaves almost as if it were a living organism. And as we’ve all experienced, we hate it as often as we’re pleased with it. The ability to predict how software will behave in the future is far from an engineering discipline. If it were, the universal certification of software would be much closer than it is. This unfortunate reality is one reason that software can be so frustrating from a user perspective.

Two conferences that bring software scholars and practitioners together—where you can pick up some of this knowledge—are worth mentioning here: the Foundations of Software Engineering (colocated this year with ACM Sigsoft 2004 in Newport Beach, Calif., in November; www.isr.uci.edu/FSE-12) and the International Conference on Software Engineering (to be held May 2005 in St. Louis, Missouri; www.cs.wustl.edu/icse05). Both are supported by the IEEE Computer Society, and both highlight speakers who are truly on the forefront of turning software engineering into engineering (with a long way to go). If you’re looking for the latest research and experimental results for your business, these venues are worth considering.

Your organization’s core business principles

Core business principles refer to those practices that a company, institution, or government agency uses to enable the organization’s viability. Whether Web services, supply chain, payroll, timesheets, or other application, there must be a relationship between your organization’s core business principles and how its software’s functionality is defined, developed, deployed, tested, and maintained.

Any person with a little knowledge can build software, but to add value to a business, the software, developers, methodologies used, metrics collected, test cases created, and test infrastructure all must align with the organization’s goals and be treated as organizational assets. The “business of software engineering” also considers the maintenance of these assets as vital to preserving the organization. And obviously, business software has to have quality built in from Day 1.

We all hear a lot of talk about “best practices” and who has the “best of the best.” Believe it or not, I see organizations every day that don’t even have “worst” software engineering practices yet. Not only are they deficient in which practices to use, but they have no idea how their software impacts their business. Any hope of developing or acquiring software that best fits their business needs will be problematic for such organizations.

Remember that software engineering is a business that runs large parts of your businesses. Be sure to treat it and its artifacts as corporate assets.

About the Author

Jeffrey Voas is a cofounder and chief scientist of Cigital. His research interests include composition strategies for COTS software, software product certification and warranties, and software quality measurement. He received his PhD in computer science from the College of William & Mary. He is a senior member of the IEEE, president of the IEEE Reliability Society, and a member of IEEE Software’s Advisory Board. Contact him at voas@cigital.com.